Incremental Learning Meets Reduced Precision Networks

Hu, Yuhuang ; Delbruck, Tobi ; Liu, Shih-Chii

Abstract: Hardware accelerators for Deep Neural Networks (DNNs) that use reduced precision parameters are more energy efficient than the equivalent full precision networks. While many studies have focused on reduced precision training methods for supervised networks with the availability of large datasets, less work has been reported on incremental learning algorithms that adapt the network for new classes and the consequence of reduced precision has on these algorithms. This paper presents an empirical study of how reduced precision training methods impact the iCARL incremental learning algorithm. The incremental network accuracies on the CIFAR-100 image dataset show that weights can be quantized to 1 bit (2.39% drop in accuracy) but when activations are quantized to 1 bit, the accuracy drops much more (12.75%). Quantizing gradients from 32 to 8 bits only affects the accuracies of the trained network by less than 1%. These results are encouraging for hardware accelerators that support incremental learning algorithms.

DOI: https://doi.org/10.1109/iscas.2019.8702541

Accepted for publication at the
2019 IEEE International Symposium on Circuits and Systems
Sapporo, Hokkaido, Japan, 2019

Incremental Learning meets Reduced Precision Networks

Yuhuang Hu, Tobi Delbruck, and Shih-Chii Liu
Institute of Neuroinformatics, University of Zürich and ETH Zürich, Zürich, Switzerland

***Abstract*—Hardware accelerators for Deep Neural Networks (DNNs) that use reduced precision parameters are more energy efficient than the equivalent full precision networks. While many studies have focused on reduced precision training methods for supervised networks with the availability of large datasets, less work has been reported on incremental learning algorithms that adapt the network for new classes and the consequence of reduced precision has on these algorithms. This paper presents an empirical study of how reduced precision training methods impact the iCARL incremental learning algorithm. The incremental network accuracies on the CIFAR-100 image dataset show that weights can be quantized to 1 bit (2.39% drop in accuracy) but when activations are quantized to 1 bit, the accuracy drops much more (12.75%). Quantizing gradients from 32 to 8 bits only affects the accuracies of the trained network by less than 1%. These results are encouraging for hardware accelerators that support incremental learning algorithms.**

I. Introduction

The circuits and systems community has recently seen rapid development of specialized hardware accelerators for implementing Deep Neural Networks (DNNs), in particular, Convolutional Neural Networks (CNNs). These systems offer better energy-efficient solutions than Graphics Processing Units (GPUs) typically used in servers. These hardware accelerators [1], [2], [3], [4] are also useful for mobile platforms with limited hardware resources. To achieve even higher energy efficiency, most accelerators employ Reduced Precision Networks (RPNs) or compressed networks. Various methods have been proposed for training a network with reduced bit precision parameters [5], [6], [7], [8] while still ensuring that the network accuracy is close to that of the Full Precision Network (FPN) during inference. The extension of these studies to training methods for reduced precision in both activations and weights; and also backpropagating gradients is presented in [5], [9], [10], [11].

Because the model size and computational complexity of RPNs are reduced compared to an FPN, memory storage and memory accesses of an RPN can also be correspondingly reduced therefore leading to reduced energy dissipation and hardware resources [12], [13].

Many studies have shown how reduced precision parameters offer better energy efficiency numbers for hardware DNN accelerators, e.g., [13], but no study has been carried out on incremental learning network algorithms that modify the parameters of a previously trained network so that the network can classify new output classes, and without the expensive retraining over all data. These algorithms attempt to maintain the accuracy of the old output classes, therefore addressing the problem known as catastrophic forgetting [14], [15]. In this paper, we study how reducing the precision of the weights, activations, and gradients during training, affect network accuracy for a particular incremental learning algorithm called iCaRL.

This work was partially supported by the Swiss National Science Foundation, HEAR-EAR, 200021_172553, the National Centre of Competence in Research (NCCR) Robotics and the Samsung Institute of Advanced Technology.

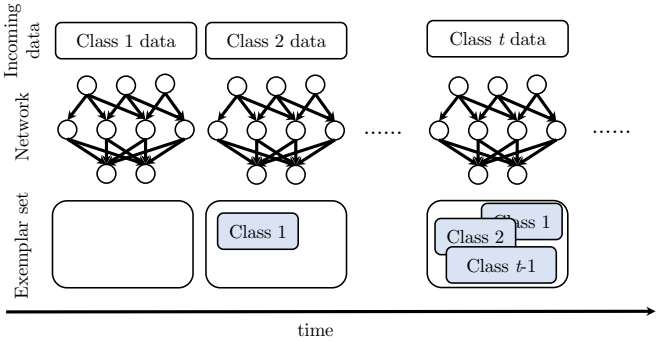

Fig. 1. Illustration of incremental learning with Reduced Precision Networks. At time t, a new batch of data is received and trained together with the previously stored exemplars. The exemplar set is updated after every incremental training session.

II. Methods

We describe the incremental learning algorithm used in this study in Section II-A and the RPN experiments in Section II-B.

A. Incremental learning algorithm

Various algorithms for overcoming the problem of catastrophic forgetting have been proposed, *e.g.*, [16], [17], [18]. The iCaRL (Incremental Classifier and Representation Learning) [19] algorithm is chosen for this study because it uses a bounded memory for storing exemplars of the previously trained or "old" classes. The algorithm maintains a subset of previous training instances to perform *prototype rehearsal* and preserves the learned classes with *knowledge distillation* [20]. We summarize iCaRL in Figs 1 and 2. For each training session, the network receives a batch of samples for the new classes. Together with the exemplars of the old classes, the network is retrained to classify both old and new classes. Because iCaRL uses a fixed-size exemplar set, the number of exemplars used for the old classes is reduced so that there is room to store exemplars for the new classes. An exemplar

is selected or removed according to the Euclidean distance between the class mean of the exemplars and the candidate samples.

Input: $X^s, \ldots, X^t$ {training examples in per-class sets}
K {memory set}
Require: Θ {current model parameters}
$\mathcal{P} = (P_1, \ldots, P_{s-1})$ {current exemplar set}
$\Theta \leftarrow$ UPDATEREPRESENTATION$(X^s, \ldots, X^t; \mathcal{P}, \Theta)$
$m \leftarrow K/t$ {number of exemplars per class}
for $y = 1, \ldots, s-1$ **do**
  $P_y \leftarrow$ REDUCEEXEMPLARSET(P_y, m)
end for
for $y = s, \ldots, t$ **do**
  $P_y \leftarrow$ CONSTRUCTEXEMPLARSET(X_y, m, Θ)
end for
$\mathcal{P} \leftarrow (P_1, \ldots, P_t)$ {new exemplar set}

Fig. 2. iCaRL incremental training algorithm [19].

B. Reduced precision networks

We implement RPNs by adopting a state-of-the-art method proposed for training Wide Reduced Precision Networks (WRPNs) [21]. This method features a quantization scheme for reducing the bit precision of both weights and activations. Although the original WRPN work did not look at the effect of reduced precision of gradients during training, we included this study here by using the gradient quantization scheme proposed in [8]. The details of the quantization methods are presented in Section II-C.

This study also includes the impact of the widening factor (the increase in the number of feature maps or hidden units) of the WRPNs on network accuracy as previously presented in [21].

C. Quantization method

The quantization function below maps a full precision floating number into the target k-bit precision (such as 4- or 8-bit precision):

$$\mathbf{r}_k = \mathcal{Q}(\mathbf{r}_{\text{in}}) = \frac{1}{2^k - 1}\text{round}\left((2^k - 1)\mathbf{r}_{\text{in}}\right) \tag{1}$$

where $\mathbf{r}_{\text{in}}$ and $\mathbf{r}_k$ are the full precision input tensor and the k-bit precision tensors respectively. The quantization is done in an element-wise manner.

The full precision weight or activation tensors are first clipped between the range of $[-1, 1]$. Then $\mathcal{Q}$ is applied to the tensors:

$$\mathbf{w}_k = \mathcal{Q}\left(\text{clip}(\mathbf{w}_{\text{in}}, [-1, 1])\right) \tag{2}$$

$$\mathbf{a}_k = \mathcal{Q}\left(\text{clip}(\mathbf{a}_{\text{in}}, [0, 1])\right) \tag{3}$$

To investigate the impact of low precision gradients during training on network accuracy, we adopted the gradient quantization mechanism from [8]. This method first maps the gradient, $\mathbf{g}_{\text{in}}$, into $[0, 1]$ and quantizes the rescaled gradient accordingly (Eq. 4). To compensate for the potential bias introduced by the quantization, an additional noise N_k term is applied (Eq. 5). Finally, Eq. 6 rescales the quantized gradient back to its original magnitude. Note that the $\max_{\bar{0}}$ takes the maximum values per sample instead of the entire gradient tensor.

$$\tilde{\mathbf{g}}_k = \mathcal{Q}\left(\text{clip}\left(\frac{\mathbf{g}_{\text{in}}}{2\max_{\bar{0}}(|\mathbf{g}_{\text{in}}|)} + \frac{1}{2} + N(k), [0, 1]\right)\right) - \frac{1}{2} \tag{4}$$

$$N_k = \frac{\sigma}{2^k - 1}; \quad \sigma \sim \text{Uniform}(-0.5, 0.5) \tag{5}$$

$$\mathbf{g}_k = 2\max_{\bar{0}}(|\mathbf{g}_{\text{in}}|)\tilde{\mathbf{g}}_k \tag{6}$$

III. RESULTS

We report on the experiments that compare the performance of the iCaRL RPNs under different bit precision settings. Section III-A presents the three datasets used in the experiments and the data preparation. Section III-B presents the implementation details and the training procedures for the RPNs and Section III-C discusses the experimental results.

A. Datasets

We chose three datasets, CIFAR-100 [22], AudioSet [15], and TinyImageNet[1], to cover different sensory modalities. These datasets have a small number of samples per class compare to common benchmark datasets such as ImageNet, so they are well suited for evaluating the performance of the incremental learning algorithm.

Dataset details are given in Table I. CIFAR-100 consists of 60,000 32×32 RGB images. AudioSet is a variant of the original audio dataset described in [23]. Each sample of AudioSet has ten seconds of audio features that are concatenated together. The features of the TinyImageNet images are extracted from a ResNet-50 [24] pretrained on ImageNet [25].

TABLE I
DATASET DESCRIPTION.

	CIFAR-100	**AudioSet**	**TinyImageNet**
Data type	RGB Image	Audio	RGB Image
Data size	32×32	1280	2048
No. classes	100	100	200
No. train samples	50,000	28,779	100,000
No. test samples	10,000	5,523	20,000
Train samples/class	500	250-300	500
Test samples/class	100	43-62	100

B. Training and implementation details

We use three different networks: ResNet-32, FCN-AS, and FCN-TIN for classifying CIFAR-100, AudioSet, and TinyImageNet respectively. These networks are described as follows:

- ResNet-32 is a 32-layer Residual Network with the same network structure and parameter configuration described in [24]. The network utilizes ReLU as the activation

[1]Source: https://tiny-imagenet.herokuapp.com/

for hidden layers and Batch Normalization (BN) for accelerating the training [26].

- FCN-AS and FCN-TIN are Multilayer Perceptron (MLP) networks with two hidden layers. The first hidden layer has 256 units and the second hidden layer has 128 units. The ReLU activation function is used. BN is applied before the activation. The inputs for both networks are features extracted from pretrained feature extractors. The output for FCN-AS has 100 units, and the output for FCN-TIN has 200 units.

As discussed in Section II-B, WRPNs increases the size of an RPN by a widening factor for achieving a comparable performance of an FPN. In this paper, we compare the performance of the original network (**1x**) to one with double the size (**2x**). The 2x networks have twice the number of feature maps or hidden units as the 1x networks. Table II summarizes the network sizes and Multiply-Accumulate (MAC) operations per input sample. With a 45 nm process as described in [27], Table II also estimates the energy consumption per input for each type of networks in 8-bit and 32-bit.

TABLE II
NUMBER OF PARAMETERS, MAC OPERATIONS PER INPUT AND ENERGY CONSUMPTION ESTIMATION PER INPUT IN 8-BIT AND 32-BIT FOR EACH TYPE OF NETWORKS.

Width	ResNet-32			
	Params	**FLOps**	**8-bit**	**32-bit**
1x	0.47M	0.138G	0.05x (15.87μJ)	1.00x (317.4μJ)
2x	1.87M	0.544G	0.20x (62.56μJ)	3.94x (1251.2μJ)
Width	**FCN-AS**			
	Params	**FLOps**	**8-bit**	**32-bit**
1x	0.44M	7.37M	0.05x (0.85μJ)	1.00x (16.95μJ)
2x	1.08M	17.6M	0.12x (2.02μJ)	2.39x (40.48μJ)
Width	**FCN-TIN**			
	Params	**FLOps**	**8-bit**	**32-bit**
1x	0.65M	11.0M	0.05x (1.27μJ)	1.00x (25.3μJ)
2x	1.50M	24.7M	0.11x (2.84μJ)	2.25x (56.81μJ)

For comparison with the iCaRL networks, the baseline networks are trained on the full dataset for 200 epochs. The training is done using the mini-batch Stochastic Gradient Descent (SGD) optimizer with a momentum of 0.9, a weight decay of 10^{-4}, and a mini-batch size of 128. The learning rate is set at 0.1 and is divided by 10 at epochs 83 and 123. This training procedure is inspired by [24].

The training schedule for iCaRL follows the procedure in [19]. The capacity of the exemplar set $\mathcal{P}$ is $N \times C$ where N is the maximum number of the exemplars retained per class and C is the total number of classes. For all experiments, we use $N = 100$. The number of training epochs for CIFAR-100, AudioSet, and TinyImageNet is 70, 40, and 40 correspondingly. The initial learning rate of 2.0 is divided by 1/5 at epochs 49, 63 for CIFAR-100 and epochs 10, 20 for AudioSet and TinyImageNet. For each incremental training session, ten random new classes are added for incremental learning, e.g., for TinyImageNet, there are a total of 20 training sessions.

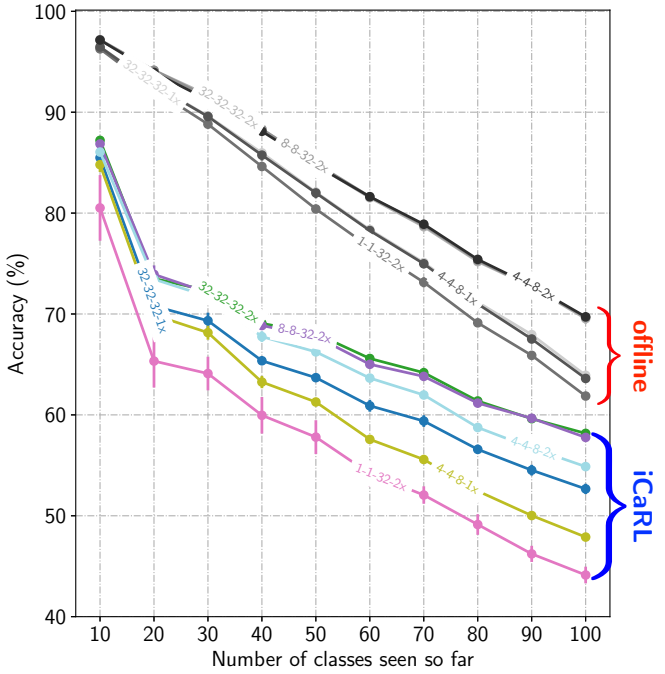

Fig. 3. Accuracy on CIFAR-100 by incrementally adding 10 new classes in each training session until all 100 classes are seen. The curves are labelled as `Weight-Activation-Gradient-Width`. The grey curves are offline baselines while the color curves are iCaRL models.

C. Discussion

Fig. 3 (discussed later) and Table III summarizes the classification accuracies for the different cases of bit precision weights, activations, and gradients. The first set of numbers are offline baseline numbers from 32-bit floating point FPNs and fixed point RPNs. The iCaRL results are presented in three groups. The first group is from the baseline FPN models where the weights, activations, and gradients are represented as 32-bit floating point numbers. The second group is for reduced precision weight and activations, but with full precision gradients. The third group uses 8-bit gradients so that we can compare the different bit precision weight and activation cases with the ones using the full precision gradient.

To assess a model's capability of acquiring new knowledge and retaining learned information, we compute a metric of average incremental accuracy introduced in [15]: $\Omega = (1/(T-1)) \sum_{t=2}^{T} \alpha_t/\alpha_{\text{offline}}$ where T is the number of training sessions, *e.g.*, $T = 10$ for CIFAR-100. α_t and α_{offline} are the classification accuracies at the training session t and for the offline baseline, respectively. Because the accuracy of the offline baseline is usually highest, Ω is usually between 0 and 1, with higher Ω signaling better incremental learning. Table III includes Ω for some of the models.

Fig. 3 shows an example of how the network's accuracy decreases after each training session for CIFAR-100. In general, the accuracy drops faster for the low precision incremental models (colored curves) compared to the full precision incremental baseline models (black curves). Although the performance gap between offline models and incremental models

TABLE III

CLASSIFICATION ACCURACY AND Ω-METRIC FOR RPN EXPERIMENTS ON ICARL. *W* - WEIGHT PRECISION, *A* - ACTIVATION PRECISION, *G* - GRADIENT PRECISION, *Width* - THE WIDENING FACTOR. STANDARD ERRORS ARE FROM FIVE REPEATS WITH RANDOM SEEDS.

W	A	G	Width	CIFAR-100		AudioSet		TinyImageNet	
				Accuracy	**Ω score**	**Accuracy**	**Ω score**	**Accuracy**	**Ω score**
				Offline baseline					
32	32	32	1x	$63.92\% \pm 0.44\%$	-	$42.61\% \pm 0.59\%$	-	$58.55\% \pm 0.29\%$	-
32	32	32	2x	$69.54\% \pm 0.50\%$	-	$43.95\% \pm 0.12\%$	-	$60.56\% \pm 0.31\%$	-
8	8	32	2x	$69.66\% \pm 0.35\%$	-	$43.99\% \pm 0.34\%$	-	$60.33\% \pm 0.24\%$	-
4	4	32	2x	$69.79\% \pm 0.26\%$	-	$43.48\% \pm 0.39\%$	-	$60.52\% \pm 0.21\%$	-
1	1	32	2x	$61.87\% \pm 0.24\%$	-	$41.98\% \pm 0.30\%$	-	$57.78\% \pm 0.40\%$	-
4	4	8	1x	$63.61\% \pm 0.29\%$	-	$42.67\% \pm 0.65\%$	-	$58.49\% \pm 0.27\%$	-
4	4	8	2x	$69.72\% \pm 0.13\%$	-	$43.91\% \pm 0.30\%$	-	$60.66\% \pm 0.14\%$	-
1	1	8	2x	$62.01\% \pm 0.13\%$	-	$41.83\% \pm 0.52\%$	-	$57.81\% \pm 0.23\%$	-
				Full precision incremental learning baseline					
32	32	32	1x	$52.66\% \pm 0.51\%$	0.785 ± 0.003	$40.56\% \pm 0.35\%$	0.742 ± 0.007	$55.95\% \pm 0.29\%$	0.864 ± 0.003
32	32	32	2x	$58.17\% \pm 0.37\%$	0.805 ± 0.004	$41.49\% \pm 0.30\%$	0.755 ± 0.009	$56.56\% \pm 0.19\%$	0.859 ± 0.001
				Low precision weight and activation (incremental learning)					
8	8	32	2x	$57.77\% \pm 0.29\%$	0.802 ± 0.003	$41.60\% \pm 0.42\%$	0.756 ± 0.008	$56.51\% \pm 0.24\%$	0.860 ± 0.003
8	4	32	2x	$57.63\% \pm 0.17\%$	-	$41.57\% \pm 0.71\%$	-	$56.41\% \pm 0.13\%$	-
8	1	32	2x	$49.88\% \pm 0.36\%$	-	$40.66\% \pm 0.41\%$	-	$\mathbf{48.05\% \pm 0.22\%}$	-
4	8	32	2x	$55.45\% \pm 0.41\%$	-	$40.81\% \pm 0.44\%$	-	$55.23\% \pm 0.23\%$	-
4	4	32	2x	$55.21\% \pm 0.15\%$	0.785 ± 0.003	$40.94\% \pm 0.18\%$	0.751 ± 0.005	$54.79\% \pm 0.27\%$	0.850 ± 0.002
4	1	32	2x	$\mathbf{42.26\% \pm 0.67\%}$	-	$40.17\% \pm 0.22\%$	-	$46.28\% \pm 0.19\%$	-
1	8	32	2x	$56.19\% \pm 0.45\%$	-	$41.14\% \pm 0.42\%$	-	$\mathbf{56.46\% \pm 0.14\%}$	-
1	4	32	2x	$\mathbf{55.35\% \pm 0.56\%}$	-	$40.67\% \pm 0.29\%$	-	$55.97\% \pm 0.19\%$	-
1	1	32	2x	$44.13\% \pm 0.82\%$	0.711 ± 0.017	$40.29\% \pm 0.31\%$	0.747 ± 0.013	$45.07\% \pm 0.23\%$	0.815 ± 0.003
				Low precision gradient (incremental learning)					
4	4	8	1x	$47.89\% \pm 0.33\%$	0.746 ± 0.007	$39.49\% \pm 0.28\%$	0.729 ± 0.009	$54.26\% \pm 0.17\%$	0.855 ± 0.006
4	4	8	2x	$54.87\% \pm 0.40\%$	0.781 ± 0.003	$41.31\% \pm 0.21\%$	0.745 ± 0.003	$54.92\% \pm 0.39\%$	0.848 ± 0.002
4	1	8	2x	$\mathbf{1.00\% \pm 0.00\%}$	-	$40.23\% \pm 0.44\%$	-	$46.53\% \pm 0.31\%$	-
1	4	8	2x	$55.81\% \pm 0.24\%$	-	$41.31\% \pm 0.40\%$	-	$56.03\% \pm 0.27\%$	-
1	1	8	2x	$\mathbf{1.00\% \pm 0.00\%}$	0.026 ± 0.000	$39.89\% \pm 0.42\%$	0.743 ± 0.012	$46.03\% \pm 0.24\%$	0.813 ± 0.002

in accuracy are similar for 1x and 2x networks (*e.g.*, around 11% for CIFAR-100), the Ω scores for 2x models are generally higher than the corresponding 1x models, which suggests that the 2x networks perform better than the 1x networks in final accuracy, and they also retain more knowledge than the thin networks. The accuracy of a network is more sensitive to the precision of its activations than that of its weights. 1-bit activation models have significantly lower accuracies than similar models that use 4-bit or higher precision activations. On the other hand, models using 1-bit weights achieve comparable accuracy with models that use the same activation and gradient precision settings. Finally, models trained using 8-bit gradient have similar accuracies to the models trained using full precision gradients.

Unlike ResNet-32 for CIFAR-100, FCN-AS and FCN-TIN are MLPs that are trained using features from a pretrained network. Therefore, only the last few layers of these networks are trained using the iCaRL algorithm. However, the performance gap between the offline and incremental models are smaller than the ResNet-32 used for CIFAR-100. One could use this hybrid method, *e.g.*, combining pretrained feature extraction layers and training only the top layers for reducing the accuracy gap between offline and incremental models.

IV. CONCLUSION

We present a study of how RPNs impact a particular incremental network algorithm called iCaRL. This study is useful for understanding a deep neural network's performance for on-device learning. The results show similar trends of results from reduced precision studies in deep networks [5], [6], [7], [8], [9], [10], [11], i.e, the network accuracies when using 1-bit activations are significantly lower than when using 1-bit weights. Our studies show that the use of 1-bit activations decreases the CIFAR-100 FP accuracies by 12.75% compared to the use of 1-bit weights (2.39%). Furthermore, the RPNs significantly reduce the energy consumption (see Table II). The ResNet-32 ASIC network would dissipate 20x less energy for an 8-bit RPN compared to that of the FCN. This study will be useful for future ASIC and FPGA deep network hardware accelerators that already consider variable precision networks [28] [29].

REFERENCES

[1] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, 2017.

[2] B. Moons and M. Verhelst, "An energy-efficient precision-scalable ConvNet processor in 40-nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 4, pp. 903–914, 2017.

[3] R. Andri, L. Cavigelli, D. Rossi, and L. Benini, "YodaNN: An architecture for ultralow power binary-weight cnn acceleration," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 1, pp. 48–60, 2018.

[4] A. Aimar, H. Mostafa, E. Calabrese, A. Rios-Navarro, R. Tapiador-Morales, I.-A. Lungu, M. B. Milde, F. Corradi, A. Linares-Barranco, S.-C. Liu, *et al.*, "Nullhop: A flexible convolutional neural network accelerator based on sparse representations of feature maps," *IEEE transactions on neural networks and learning systems*, no. 99, pp. 1–13, 2018.

[5] M. Courbariaux, Y. Bengio, and J.-P. David, "BinaryConnect: Training deep neural networks with binary weights during propagations," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 3123–3131.

[6] E. Stromatias, D. Neil, M. Pfeiffer, F. Galluppi, S. B. Furber, and S.-C. Liu, "Robustness of spiking deep belief networks to noise and reduced bit precision of neuro-inspired hardware platforms," *Frontiers in neuroscience*, vol. 9, p. 222, 2015.

[7] J. Ott, Z. Lin, Y. Zhang, S. Liu, and Y. Bengio, "Recurrent neural networks with limited numerical precision," *CoRR*, vol. abs/1608.06902, 2016.

[8] S. Zhou, Z. Ni, X. Zhou, H. Wen, Y. Wu, and Y. Zou, "DoReFa-Net: Training low bitwidth convolutional neural networks with low bitwidth gradients," *CoRR*, vol. abs/1606.06160, 2016.

[9] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 4107–4115.

[10] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet classification using binary convolutional neural networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 525–542.

[11] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *Journal of Machine Learning Research*, vol. 18, no. 187, pp. 1–30, 2018.

[12] M. Horowitz, "Energy table for 45nm process." [Online]. Available: Available:https://sites.google.com/site/seecproject

[13] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, "EIE: efficient inference engine on compressed deep neural network," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2016, pp. 243–254.

[14] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in *Psychology of Learning and Motivation*, G. H. Bower, Ed. Academic Press, 1989, vol. 24, pp. 109 – 165.

[15] R. Kemker, M. McClure, A. Abitino, T. Hayes, and C. Kanan, "Measuring catastrophic forgetting in neural networks," in *AAAI Conference on Artificial Intelligence*, 2018.

[16] A. Gepperth and C. Karaoguz, "A Bio-Inspired Incremental Learning Architecture for Applied Perceptual Problems," *Cognitive Computation*, vol. 8, pp. 924 – 934, 2016.

[17] Z. Li and D. Hoiem, "Learning without forgetting," in *European Conference on Computer Vision*. Springer, 2016, pp. 614–629.

[18] R. Kemker and C. Kanan, "Fearnet: Brain-inspired model for incremental learning," in *International Conference on Learning Representations*, 2018.

[19] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "iCaRL: Incremental classifier and representation learning," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[20] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Deep Learning and Representation Learning Workshop: NIPS 2014*, 2014.

[21] A. Mishra, E. Nurvitadhi, J. J. Cook, and D. Marr, "WRPN: Wide reduced-precision networks," in *International Conference on Learning Representations*, 2018.

[22] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," University of Toronto, Tech. Rep., 2009.

[23] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio Set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE ICASSP 2017*, New Orleans, LA, 2017.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[25] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 248–255.

[26] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 448–456.

[27] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2014, pp. 10–14.

[28] B. Moons and M. Verhelst, "A 0.32.6 TOPS/W precision-scalable processor for real-time large-scale ConvNets," in *2016 IEEE Symposium on VLSI Circuits*, June 2016, pp. 1–2.

[29] D. Shin, J. Lee, J. Lee, J. Lee, and H. Yoo, "DNPU: An energy-efficient deep-learning processor with heterogeneous multi-core architecture," *IEEE Micro*, vol. 38, no. 5, pp. 85–93, Sep 2018.